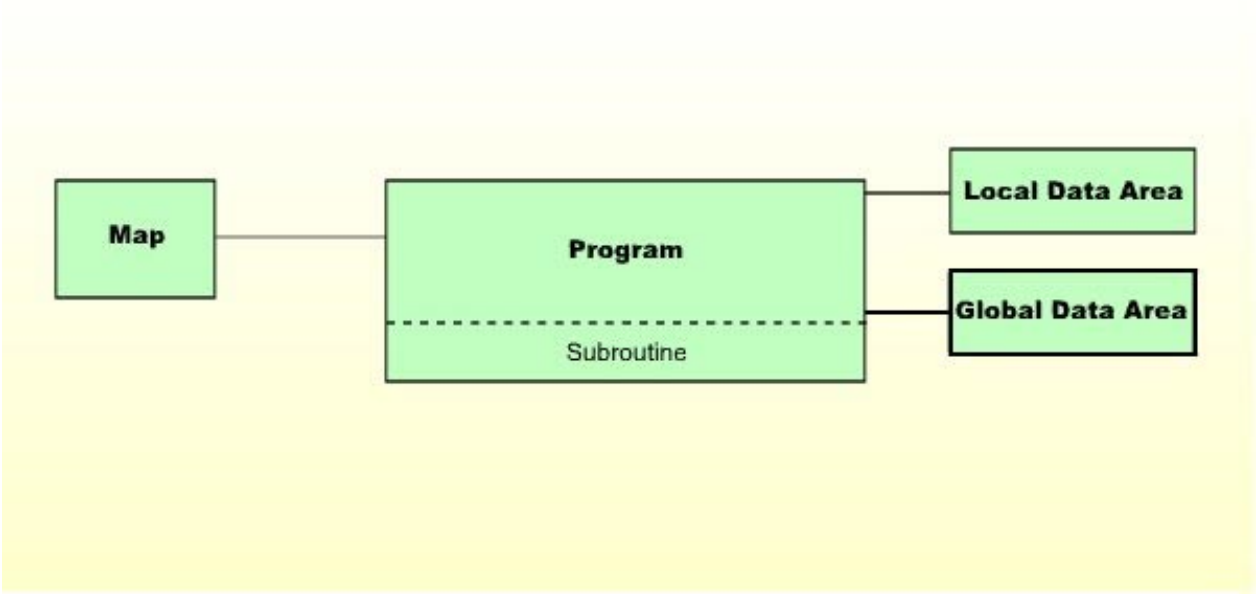
Session 3 - Creating a Global Data Area

In Natural, data can be defined in a single location outside any particular program or routine. Data defined in such a global data area can then be shared by multiple programs/routines.

In this session, you will create a global data area. In addition, the local data area created in Session 2 will be modified. In addition, the program has to be modified to reference not only the local data area, but also the new global data area.



Listed below are Steps 1 to 7 of Session 3.

Step 1

On the Development Functions menu, enter the code **E** and the name LDA01. This invokes the data area editor and reads the local data area LDA01 into the work area of the editor.

Step 2

To create the new data area, save the contents of the work area under a different name: in the Command line of the editor, enter the command SAVE GDA01.

Then enter the command READ GDA01 to read the newly created data area into the work area. As it is identical to the original one, only the name displayed at the top of the data area editor will change from LDA01 to GDA01.

Step 3

Then enter the command SET TYPE GLOBAL. As a result, you will now be editing the data area in the editor as a global data area.

Enter the line command **.D** to delete the first two lines (#NAME-START and #NAME-END). The data area now looks as follows:

```
Global    GDA01     Library SYSTEM                              DBID  10 FNR  32
Command                                                                 > +
I T L Name                            F Leng Index/Init/EM/Name/Comment
All - ------------------------------- - ---- --------------------------------
    1 #MARK                           A    1
  V 1 EMPLOYEES-VIEW                             EMPLOYEES
    2 PERSONNEL-ID                    A    8
    2 NAME                            A   20
    2 DEPT                            A    6
    2 LEAVE-DUE                       N  2.0
                                                                 S 6    L 1
```

Keep in mind that a data area must be compiled and stored in object form before any program referencing that data area can be compiled and executed.

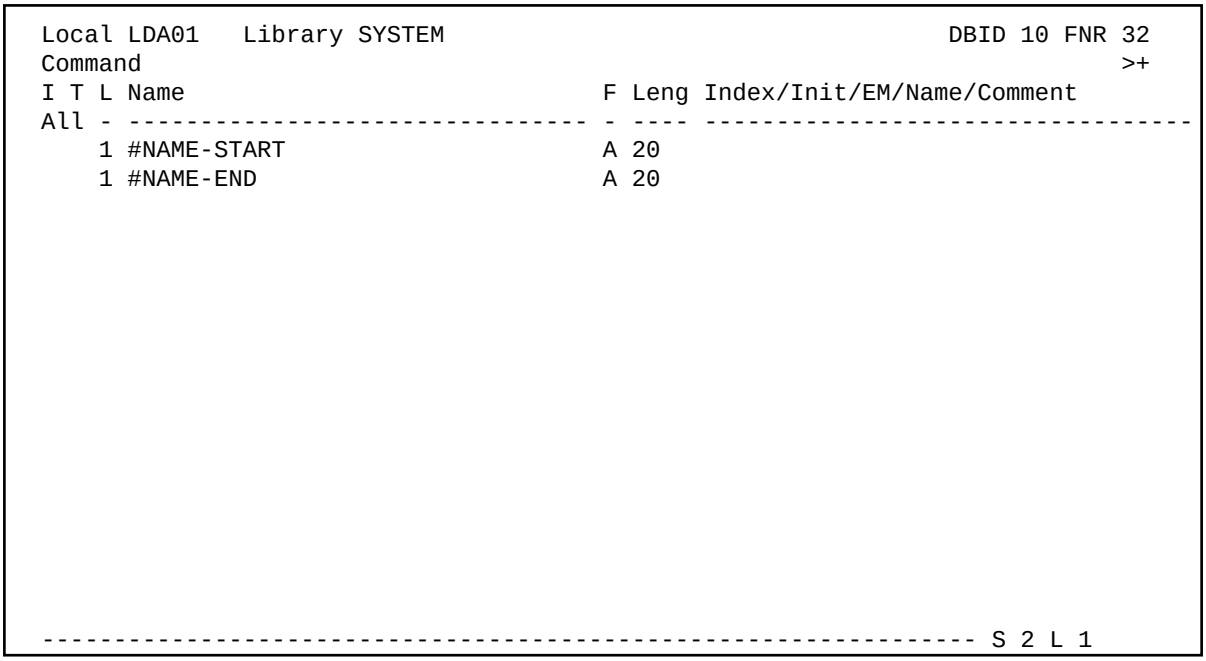
In the Command line, enter the command STOW. GDA01 is now compiled and stored in source and object form.

Step 4

Now some variables must be removed from the local data area, because they are now defined in the new global data area.

In the Command line, enter the command READ LDA01 to read the local data area LDA01 into the editor.

With the line command **.D**, delete from LDA01 all variables that are now also defined in GDA01; that is, everything except the two variables #NAME-START and #NAME-END. The modified data area now looks as follows:



In the Command line, enter the command STOW.

The modified local data area is now ready to be referenced in the program.

Step 5

In the Command line of the data area editor, enter the command EDIT PGM01. As PGM01 is stored as an object of type program, the program editor will automatically be invoked.

As the defined data are now located in two data areas, the DEFINE DATA statement in the program must now reference the global data area GDA01 as well as the local data area LDA01.

In the line that contains DEFINE DATA, enter the command **.I(1)** to insert one empty line. Type in GLOBAL USING GDA01 in the empty line.

The DEFINE DATA statement block should now look as follows:

```
...
DEFINE DATA
  GLOBAL USING GDA01
  LOCAL  USING LDA01
END-DEFINE
...
```

Step 6

In addition, we will change the instructions for the output produced by the program: we will add a WRITE TITLE statement and modify the DISPLAY statement.

Using the program example below, modify the program to include the WRITE TITLE statement (above the DISPLAY statement) and the new format in the DISPLAY statement. Use the line command **.I** to create the empty lines you need for these modifications. Also, add some comments at the top of the program to indicate the changes you have made.

The WRITE TITLE statement used in this program produces a multiple-line title in the resulting report. The slash (/) notation causes a line advance. As nothing else is specified, the title lines will be displayed centered and not underlined.

The revised program - particularly the DEFINE DATA, WRITE TITLE and DISPLAY statement blocks - should now look as follows:

Program PGM01:

```
* Example Program 'PGM01' for Natural Tutorial
* PROGRAM NOW USES A LOCAL DATA AREA
* A GLOBAL DATA AREA AND TITLE HAVE BEEN ADDED AND
*   THE DISPLAY STATEMENT HAS BEEN CHANGED
* ------------------------------------------------
DEFINE DATA
  GLOBAL USING GDA01
  LOCAL  USING LDA01
END-DEFINE                                          *
REPEAT
*
  INPUT USING MAP 'MAP01'
  IF #NAME-START = '.'
     ESCAPE BOTTOM
  END-IF
  MOVE #NAME-START  TO  #NAME-END
*
  RD1. READ EMPLOYEES-VIEW BY NAME
         STARTING FROM #NAME-START
         THRU #NAME-END
     IF LEAVE-DUE >= 20
       PERFORM MARK-SPECIAL-EMPLOYEES
     ELSE
       RESET #MARK
     END-IF
*
WRITE TITLE / '*** PERSONS WITH 20 OR MORE DAYS LEAVE DUE ***'
            / '***     ARE MARKED WITH AN ASTERISK        ***'// *
  DISPLAY 23X '//N A M E' NAME
           3X '//DEPT'    DEPT
           3X '//LV/DUE'  LEAVE-DUE
           3X '//*'       #MARK          *
  END-READ
*
  IF *COUNTER (RD1.) = 0
    REINPUT 'PLEASE TRY ANOTHER NAME'
  END-IF
*
END-REPEAT
*
DEFINE SUBROUTINE MARK-SPECIAL-EMPLOYEES
    MOVE '*' TO #MARK
END-SUBROUTINE
END
```

Step 7

Once you have completed all changes, CHECK the program and correct any errors that may have occurred.

Then RUN the program. On the input screen, enter the name JONES as starting name.

Note the differences in the report output, which should now look as follows:

```
*** PERSONS WITH 20 OR MORE DAYS LEAVE DUE ***
***      ARE MARKED WITH AN ASTERISK       ***



          N A M E           DEPT    LV    *
                                    DUE
     --------------------   ------   ---   -

     JONES                  SALE30    25   *
     JONES                  MGMT10    34   *
     JONES                  TECH10    11
     JONES                  MGMT10    18
     JONES                  TECH10    21   *
     JONES                  SALE00    30   *
     JONES                  SALE20    14
     JONES                  COMP12    26   *
     JONES                  TECH02    25   *
```

When the execution of the program has finished without any errors, STOW the program for future modification in Session 4.

End of Session 3.

